Several link keys are better than one, or extracting disjunctions of link key candidates

Manuel Atencia, Jérôme David, Jérôme Euzenat

Several Link Keys Are Better than One, or Extracting Disjunctions of Link Key Candidates

Manuel Atencia Jérôme David Jérôme Euzenat

Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, F-38000 Grenoble, France
Firstname.Lastname@inria.fr

Abstract

Link keys express conditions under which instances of two classes of different RDF data sets may be considered as equal. As such, they can be used for data interlinking. There exist algorithms to extract link key candidates from RDF data sets and different measures have been defined to evaluate the quality of link key candidates individually. For certain data sets, however, it may be necessary to use more than one link key on a pair of classes to retrieve a more complete set of links. To this end, in this paper, we define disjunction of link keys, propose strategies to extract disjunctions of link key candidates from RDF data, and apply existing quality measures to evaluate them. We also report on experiments with these strategies.

1 Introduction

Finding links across linked open data sets is an important task as it enables data interoperability. Different approaches to data interlinking have been proposed [Ferrara et al. 2011, Nentwig et al. 2017]. One of the these is based on link keys [Atencia et al. 2014]. Link keys generalise relational keys to the case of two RDF data sets. They express conditions under which two instances of different data sets may be linked. Algorithms have been proposed for extracting link key candidates from RDF data and supervised and unsupervised measures for selecting the best ones [Atencia et al. 2014, Atencia et al. 2019].

One single link key, however, even the best one, may not be enough to discover all links in certain data sets. This is simply the case of data sources covering different concepts. This may also be useful if the data related to a particular class is the result of aggregating different sources that use different properties: there may be several different ways to generate links. Thus, instead of selecting one single best link key candidate, it could be worth selecting the best combination of link key candidates, as it is already done for other link specifications [Sherif et al. 2017].

This paper addresses the specific problem of extracting boolean combinations of link key candidates from two RDF data sets. We define conjunction and disjunction of link keys in terms of their generated links. Then, we show that conjunction does not generate any link that single link key candidates do not generate already. So we focus on extraction of disjunctions of link key candidates. This is challenging because of the large number of non redundant disjunctions of link key candidates (potentially 2^n where n is the number of link key candidates).

More specifically, the main contributions of this paper are:

- The identification and precise definition of the conjunction and disjunction of link keys through their semantics and relations with other link keys,
- The extension of available link key evaluation measures to disjunctions of link keys,
- Strategies for extracting best ranked disjunctions of link key candidates based on exploiting antichains in a lattice, and
- Evaluation of these strategies experimentally.

In the following, after presenting the related work, and especially the approaches that consider combinations of link specifications (Section 2), we present link keys (Section 3). Then, we introduce conjunction and disjunction of link keys, and extend the available quality measures to evaluate disjunctions of link key candidates (Section 4). We discuss strategies for extracting disjunctions of link key candidates (Section 5) and report on efforts to extract them from several data sets (Section 6).

2 Related work

Data interlinking refers to the process of finding pairs of IRIs in two different RDF data sets that represent the same resource [Ferrara et al. 2011, Nentwig et al. 2017]. The result of this process is a set of links, which may be added to the data sets by relating the corresponding IRIs with the owl:sameAs property. Data interlinking can be defined as follows: given two sets of individual identifiers I_D and $I_{D'}$ from two data sets D and D', find the set L of pairs of identifiers $\langle o, o' \rangle \in I_D \times I_{D'}$ such that o and o' represent the same resource.

Links are usually produced by using a framework, such as SILK [Volz et al. 2009] or LIMES [Ngonga Ngomo and Auer 2011], processing *link specifications*. Link specifications indicate the conditions for two IRIs to be linked. They may be directly defined by users or (semi-)automatically extracted. This paper is concerned with the combination and evaluation of several link specifications together.

Most methods roughly compute a *numerical specification* $\langle \sigma, \theta \rangle$ made up of a similarity measure σ between the entities to be linked and a threshold θ. It is assumed that, if two entities are very similar, then they are likely the same. Hence,

such specifications generate links through (adapted from [Sherif et al. 2017]):

$$L_{\sigma,\theta}^{D,D'} = \{\langle o, o' \rangle \in I_D \times I_{D'}; \sigma(o, o') \geq \theta\}$$

WOMBAT [Sherif et al. 2017] provides a way of exploring the space of such link specifications, starting with atomic similarity between pairs of datatype property values. It is able to learn conjunction, disjunction and difference of link specifications in a supervised manner. SILK, via the ActiveGenLink algorithm [Isele and Bizer 2013], composes similarity components (similarity metrics), but not full link specifications as it is done in WOMBAT.

In this paper, we address the extraction of disjunctions of link key candidates only, as conjunction does not produce any new link that the link key candidates extracted by current extraction algorithms do not generate. Unlike WOMBAT and SILK, our approach is fully non supervised: it does not need any sample links.

Logical link specifications are logical axioms from which links are inferred. Unlike numerical specifications, they can be combined with other kinds of knowledge, such as ontologies and ontology alignments, to infer links by using reasoning [Saïs et al. 2007, Al-Bakri et al. 2015, Al-Bakri et al. 2016, Hogan et al. 2012]. Logical link specifications do not incorporate similarity metrics natively but such metrics, if necessary, may be handled separately through specific rules or in a data preprocessing step.

Key-based specifications fall into the category of logical link specifications. Key-based approaches typically extract keys from RDF data sets and combine them with ontology alignments for interlinking [Symeonidou et al. 2014, Achichi et al. 2016, Farah et al. 2017, Atencia et al. 2012].

This paper deals with *link keys*, a specific type of logical link specification. Link keys generalise keys to the case of two different RDF data sets [Euzenat and Shvaiko 2013, Atencia et al. 2014]. An example of a link key is:

$$\{\langle \mathsf{auteur}, \mathsf{creator} \rangle\}\{\langle \mathsf{titre}, \mathsf{title} \rangle\} \text{ linkkey } \langle \mathsf{Livre}, \mathsf{Book} \rangle$$

stating that whenever an instance of the class Livre has the same values for the property auteur as an instance of the class Book has for the property creator and they share at least one value for their properties titre and title, then they denote the same entity. A link key may be thought of as a pair of aligned keys, but the relation between link keys and keys is more subtle, as we explain below.

The key-based approaches proposed in [Achichi et al. 2016, Farah et al. 2017] aim at using a key extraction algorithm [Symeonidou et al. 2014] to extract pairs of keys that can be used as link specifications. They extract IN-keys, i.e. keys based on sharing one value between properties, which hold in both source and target data sets. It is assumed that both data sets are described using the same ontology or, more precisely, the system only looks for keys based on the vocabulary common to the two data sets. In this case, the extracted IN-keys, though not equal, mostly correspond to strong IN-link keys (i.e. link keys that are made up of keys), and not to weak IN-link keys, which are more general and are the kind of link keys extracted in [Atencia et al. 2014].

KeyRanker [Farah et al. 2017] describes a method for selecting and assembling a disjunction of several such pairs of keys. The selection always starts with the most covering candidate but further key pairs are selected based on the marginal covering contribution of the added candidate.

There is no necessary correspondence between keys and link keys: there might be keys unrelated to any link key, and link keys unrelated to any key [Euzenat and Shvaiko 2013, Example 5.38, p116] and [Atencia et al. 2019]. Hence, searching for keys to be eventually turned into link keys may fail.

A technique for directly extracting link keys between two classes from two RDF data sets has been proposed [Atencia et al. 2014]. Unlike key-based approaches, it does not require as input any alignment between the properties of both data sets nor makes the assumption of common vocabularies, and it avoids the generation of keys that are specific to one data set only. It first extracts link key candidates from the data and then uses measures of the quality of these candidates in order to select the one to apply. Either supervised or non supervised measures may be used.

However, the combination of link keys has not been studied yet. This paper deals with a particular way of combining link key candidates via disjunction.

3 Preliminaries

We introduce here technical notions that are necessary to make precise the different points discussed in the paper. We deal with data expressed in RDF. Each RDF data set D is a set of triples expressed with respect to its signature $\langle R_D, P_D, C_D\rangle$ in which R_D is the set of object property identifiers, P_D the set of datatype property identifiers and C_D the set of class identifiers. Moreover, I_D and L_D will denote, respectively, the set of individuals and the set of literals in D. The terms "class", "datatype property", "object property" and "individual" are used according to their meaning in RDFS and OWL.

Given $c \in C_D$, we denote by $c^D = \{t \in I_D; \langle t, \texttt{rdf:type}, c\rangle \in D\}$ the set of instances of c in the data set D. In RDF, an individual may have several different values for the same property. Hence, given a datatype property $p \in P_D$ and an individual $o \in I_D$, we denote by $p^D(o) = \{v \in L_D; \langle o, p, v\rangle \in D\}$ the set of values of property p for object o in the data set D. Similarly, given an object property $r \in R_D$, we have $r^D(o) = \{u \in I_D; \langle o, r, u\rangle \in D\}$.

[RDF Data set] Let us consider the two very simple data sets of Table 1.[1]

This type of example may occur when the second data set (D') is the result of the merge of two heterogeneous data sources (one using birthdate property and the other using building property). The signature of D is $\langle\{\}, \{$prénom, datenaiss, post, bât.$\}, \{$Employés$\}\rangle$; that of D' is $\langle\{\}, \{$firstname, birthdate, position, building$\}, \{$Staff$\}\rangle$.

[1] For the sake of readability, we represent these RDF data sets as simple relational tables without multiple values or object references. They are only here to explain the problems and this makes it easier to compare them.

D (Employés)					D' (Staff)				
id	prenom	datenaiss	poste	bât.	firstname	birthdate	position	building	id
i_2	Paul	1967	Dir.	B2	Paul		Dir.	B2	z_2
i_3	Mary	1963	Dir.	B1	Mary		Dir.	B1	z_3
i_4	John	1963	Pr.	B1	John		Pr.	B1	z_4
i_6	Bill	1980	Pr.	B1	William	1980	Pr.		z_6
i_7	Ana	1947	Dir.	B2	Ana	1947	Dir.		z_7
i_8	John	1967	Pr.	B2	John	1967	Pr.		z_8

Table 1: These two tables display instances of the classes Employés and Staff of two RDF data sets D and D'.

Link keys specify the pairs of properties to compare for deciding whether individuals of two classes of two different data sets have to be linked. We first give the definition of a link key expression.

[Link key expression [Atencia et al. 2019]] A *link key expression* over two signatures $\langle R, P, C\rangle$ and $\langle R', P', C'\rangle$ is an element of the set $2^{(P\times P')\cup(R\times R')} \times 2^{(P\times P')\cup(R\times R')} \times (C \times C')$, i.e.

$$\langle \{\langle p_i, p'_i\rangle\}_{i\in EQ}, \{\langle q_j, q'_j\rangle\}_{j\in IN}, \langle c, c'\rangle\rangle$$

such that EQ and IN are (possibly empty) finite sets of indices.

[Link key expressions] Consider the two signatures of the data sets D and D' of Example 3. The following are examples of link key expressions:

$$\begin{aligned} k &= \langle \{\langle \mathsf{datenaiss}, \mathsf{birthdate}\rangle\}, \{\}, \langle \mathsf{Employe}, \mathsf{Staff}\rangle\rangle \\ h &= \langle \{\langle \mathsf{datenaiss}, \mathsf{birthdate}\rangle\}, \{\langle \mathsf{poste}, \mathsf{position}\rangle\}, \langle \mathsf{Employe}, \mathsf{Staff}\rangle\rangle \\ l &= \langle \{\langle \mathsf{datenaiss}, \mathsf{birthdate}\rangle, \langle \mathsf{poste}, \mathsf{position}\rangle\}, \{\langle \mathsf{poste}, \mathsf{position}\rangle\}, \\ &\quad \langle \mathsf{Employe}, \mathsf{Staff}\rangle\rangle \end{aligned}$$

Link key expressions may be used to generate links between RDF data sets.

[Link set generated by a link key expression [Atencia et al. 2019]] Let D and D' be two data sets and let $k = \langle \{\langle p_i, p'_i\rangle\}_{i\in EQ}, \{\langle q_j, q'_j\rangle\}_{j\in IN}, \langle c, c'\rangle\rangle$ be a link key expression over their signatures. The *link set generated by* k for D and D' is the subset $L_k^{D,D'} \subseteq c^D \times c'^{D'}$ defined as:

$$\langle o, o'\rangle \in L_k^{D,D'} \text{ iff } \begin{cases} p_i^D(o) = p_i'^{D'}(o') & \text{for all } i \in EQ, \text{ and} \\ q_j^D(o) \cap q_j'^{D'}(o') \neq \emptyset & \text{for all } j \in IN \end{cases}$$

[Generated link sets] Given the link key expressions of Example 3 and the data sets D and D' of Example 3, the generated links are:

$$\begin{aligned} L_k^{D,D'} &= \{\langle i_7, z_7\rangle, \langle i_8, z_8\rangle, \langle i_6, z_6\rangle, \langle i_2, z_8\rangle\} \\ L_l^{D,D'} = L_h^{D,D'} &= \{\langle i_7, z_7\rangle, \langle i_8, z_8\rangle, \langle i_6, z_6\rangle\} \end{aligned}$$

The meaning of k is the set of all pairs of Employés/Staff members having exactly the same datenaiss/birthdate. That of l and h is those who, in addition, share at least one poste/position. l and h generate the same links for D and D' because the data sets are simple. If Ana (z_7) also held the position of dean, for example, then $\langle i_7, z_7\rangle$ would be in $L_h^{D,D'}$ but not in $L_l^{D,D'}$.

Link key expressions may be related by subsumption. Furthermore, the meet and join of two link key expressions can be defined.

[Subsumption, meet and join of link key expressions [Atencia et al. 2019]] Let $k = \langle E, I, \langle c, c'\rangle\rangle$ and $h = \langle F, J, \langle c, c'\rangle\rangle$ be two link key expressions with the same pair of classes $\langle c, c'\rangle$ and over the same pair of signatures $\langle R, P, C\rangle$ and $\langle R', P', C'\rangle$. We will say that k *is (intentionally) subsumed by* h, written $k \trianglelefteq h$, if $E \subseteq F$ and $I \subseteq J$. In addition, the *meet* and *join* of k and h, denoted by $k \triangle h$ and $k \triangledown h$, are defined as follows:

$$k \triangle h = \langle E \cap F, I \cap J, \langle c, c'\rangle\rangle$$
$$k \triangledown h = \langle E \cup F, I \cup J, \langle c, c'\rangle\rangle$$

Subsumption of link keys expressions is contravariant with the inclusion of their generated link sets. This reflects the fact that the more constraints there are, the less links satisfy them.

In the following, we will use the notion of extended subsumption instead of intentional subsumption. [Extended subsumption of link key expressions] Let D and D' be two data sets. Let k and h be two link key expressions over their signatures. We say that h *(extensively) subsumes* k, written $k \preceq^{D,D'} h$, if $L_k^{D,D'} \supseteq L_h^{D,D'}$. The (extensive) subsumption is thus an extension of subsumption. This is stated in Property 1 below, which can be easily proven.

Property 1 *If $k \trianglelefteq h$, then, for any data sets D, D', $k \preceq^{D,D'} h$.*

In the following, the "D, D'" exponent may be avoided since we only compare link key expressions on the same data sets. We will write $k \simeq h$ if $k \preceq h$ and $h \preceq k$, i.e. if $L_k = L_h$, and say that k and h are equivalent.

We have proposed a procedure to extract a small subset of link key expressions, called *link key candidates* [Atencia et al. 2014]. This procedure has been recast in the setting of formal concept analysis [Ganter and Wille 1999] and we have proved that the set of link key candidates K with $\trianglelefteq$ form a (concept) lattice, denoted by $\langle K, \trianglelefteq\rangle$ [Atencia et al. 2019]. The lattice for the two data sets of Example 3 is depicted in Figure 1. These are the link key candidates that we want to combine and evaluate.

In order to select the best link key candidate, discriminability and coverage measures have been proposed [Atencia et al. 2014]. These are unsupervised measures, i.e. they do not require any link as input. Below we reformulate discriminability and coverage directly with respect to the generated link sets as it will be of help later in the paper.

If L is a set of links between D and D', then we define

$$\pi(L) = \{o \in D; \langle o, o'\rangle \in L\} \text{ and } \pi'(L) = \{o' \in D'; \langle o, o'\rangle \in L\}$$

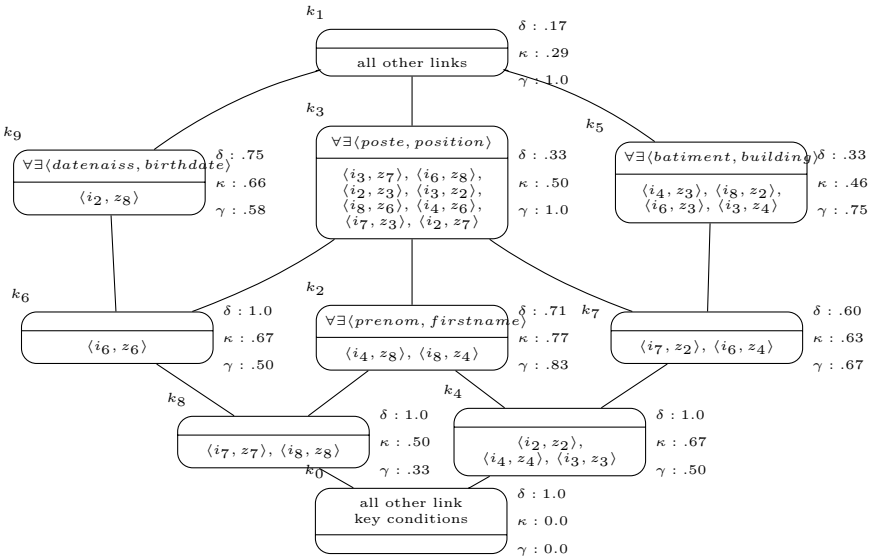

Figure 1: Lattice $\langle K, \trianglelefteq \rangle$ of extracted link key candidates from the data sets D and D' of Example 3 (δ=discriminability, γ=coverage, κ=harmonic mean between them). The notation $\forall\exists$ means that the pair of properties is both in the conditions indexed by IN and EQ. The lattice is drawn according to the conventions of formal concept analysis [Ganter and Wille 1999].

Discriminability measures how close the links generated by a link key candidate are from a one-to-one mapping. The idea behind discriminability is to disqualify link key candidates that would link the same entity of one data set to several entities from the other data set (e.g. $\langle\{\langle \mathsf{firstName}, \mathsf{givenName}\rangle\}, \{\}, \langle \mathsf{Person}, \mathsf{Person}\rangle\rangle$). Such link keys show a low discriminability on data set for which they generate such links.

[Discriminability] Let k be a link key expression,

$$\delta^{D,D'}(k) = \begin{cases} 1.0 & \text{if } L_k^{D,D'} = \varnothing \\ \dfrac{\min(|\pi(L_k^{D,D'})|, |\pi'(L_k^{D,D'})|)}{|L_k^{D,D'}|} & \text{otherwise} \end{cases}$$

Coverage measures how complete a link key candidate is with respect to the data sets, i.e. the proportion of instances of both classes that would be linked by the link key candidate: [Coverage] Let $k = \langle E, I, \langle c, c'\rangle\rangle$ be a link key expression,

$$\gamma^{D,D'}(k) = \begin{cases} 1.0 & \text{if } c^D = c'^{D'} = \varnothing \\ \dfrac{|\pi(L_k^{D,D'}) \cup \pi'(L_k^{D,D'})|}{|c^D \cup c'^{D'}|} & \text{otherwise} \end{cases}$$

The coverage measure always favours the most general link key expressions. This is stated in Property 2 below, which can be easily proven too.

Property 2 *If $h \preceq^{D,D'} k$, then $\gamma^{D,D'}(h) \geq \gamma^{D,D'}(k)$*

Using both coverage and discriminability measures strikes a balance between the completeness and generality of link key candidates. They can be aggregated by harmonic mean, here denoted by $\kappa^{D,D'}$, just like F-measure does with precision and recall (see Figure 1).

4 Conjunction and Disjunction of link keys

As explained in the previous section, the link key candidate extraction process provides a link key candidate lattice, and for each candidate it is possible to compute coverage and discriminability. The question to address is to find a combination of such link key candidates whose overall evaluation measure is higher than these. In this paper, we focus on conjunction and disjunction of link keys.

[Conjunction and disjunction of link key expressions] Given two data signatures S and S' and two link key expressions k and h over S and S', the conjunction and disjunction of k and h are denoted by $k \wedge h$ and $k \vee h$, respectively. By extension, given a finite number of link key expressions $k_1, \ldots, k_n$, the conjunction and disjunction of $k_1, \ldots, k_n$ are denoted by $k_1 \wedge \cdots \wedge k_n$ and $k_1 \vee \cdots \vee k_n$, respectively.

The semantics of the conjunction and disjunction operators are defined by the links they generate: [Link sets generated by the conjunction and the disjunction of link key expressions] Let D and D' be two data sets of signatures S and S'. Let k and h be link key expressions over S and S'. The *link sets generated by* $k \wedge h$ and $k \vee h$ are the subsets of $c^D \times c'^{D'}$ defined by

$$L_{k \wedge h}^{D,D'} = L_k^{D,D'} \cap L_h^{D,D'}$$
$$L_{k \vee h}^{D,D'} = L_k^{D,D'} \cup L_h^{D,D'}$$

Property 3 *$\wedge$ and $\vee$ are commutative and associative.*

[Conjunction and disjunction of link key expressions and their generated link sets] Taking into account the link key candidates of Figure 1, one can compute that:

$$L_{k_9 \wedge k_5}^{D,D'} = L_{k_0}^{D,D'} = \emptyset$$
$$L_{k_4 \vee k_6}^{D,D'} = \{\langle i_2, z_2\rangle, \langle i_3, z_3\rangle, \langle i_4, z_4\rangle, \langle i_6, z_6\rangle, \langle i_7, z_7\rangle, \langle i_8, z_8\rangle\}$$

Notice that the second link set, unlike the first one, is not generated by any link key candidate.

4.1 Relations between conjunctions and disjunctions of link keys

Conjunctions and disjunctions of link key expressions may in some cases be reduced to other link keys. Straightforwardly:

Property 4 *If $k \preceq h$, then $k \vee h \simeq k$ and $k \wedge h \simeq h$*

The links generated by conjunction ($\wedge$) are the same as those generated by join ($\triangledown$). This could make one think that conjunction is redundant. However, the set of link key candidates is not closed by $\triangledown$: if k and h are link key expressions, then, by definition, $k \triangledown h$ is a link key expression, but, if k and h are link key candidates, $k \triangledown h$ is not necessarily a link key candidate. Nevertheless, there is always a link key candidate that generates the same links (i.e. equivalent through $\simeq$):

Property 5 *If k and h are link key candidates, then there exists a link key candidate l such that $k \wedge h \simeq l$.*

Let k and h be two link key candidates. Let $l = k \triangledown h$. Then l is a link key candidate (i.e. the greatest common subsumee of k and h in $\langle K, \trianglelefteq \rangle$). From [Atencia et al. 2019, Lemma 1(3)], $L_l = L_{k \triangledown h} = L_k \cap L_h$. By definition, $L_{k \wedge h} = L_k \cap L_h$. Thus, $L_{k \wedge h} = L_l$, i.e. $k \wedge h \simeq l$.

Property 5 means that the conjunction of link key candidates will not bring any new link: an already available link key candidate will generate the same link set. For example, in Figure 1, $k_9 \nabla k_5$ (i.e. the link key expression made up of the union of the properties of k_9 and k_5) is not featured (i.e. it is not a link key candidate). The lowest common subsumer of k_9 and k_5 with respect to $\trianglelefteq$ (or the greatest common subsumee with respect to $\preceq$) is k_0, which is such that $k_9 \nabla k_5 \trianglelefteq k_0$ and $L_{k_9 \wedge k_5} = L_{k_0}$, or, what is the same, $k_9 \wedge k_5 \simeq k_0$.

On the contrary, the links generated by disjunction ($\vee$) are not the same as the links generated by meet ($\triangle$). The set of link key candidates is closed by $\triangle$ [Atencia et al. 2019, Lemma 3], but meet is *not* disjunction since $L_{k \triangle h} \supseteq L_k \cup L_h = L_{k \vee h}$ [Atencia et al. 2019, Lemma 1(2)]. This is exemplified in Example 4 by $k_3 = k_6 \triangle k_7$ and $L_{k_3} \supset L_{k_6} \cup L_{k_7}$.

Therefore, from here on, we focus on disjunction of link keys.

The extended subsumption relation ($\preceq$) can be straightforwardly extended to disjunction of link key expressions. Property 6 states the relations between disjunction, meet and join of link key expressions with respect to $\preceq$. This is illustrated in Figure 2.

Property 6 *If k and h are two link key expressions, then*

$$k \triangle h \preceq k \vee h \preceq k \preceq k \nabla h$$

Let k and h be link key expressions. From [Atencia et al. 2019, Lemma 1(2)] we have $L_{k \triangle h} \supseteq L_k \cup L_h$. By definition, $L_{k \vee h} = L_k \cup L_h$. Then $L_{k \triangle h} \supseteq L_{k \vee h}$, i.e. $k \triangle h \preceq k \vee h$, and $L_{k \vee h} \supseteq L_k$, i.e. $k \vee h \preceq k$. From [Atencia et al. 2019, Lemma 1(3)], $L_{k \nabla h} = L_k \cap L_h$. So $L_{k \nabla h} \subseteq L_k$, i.e. $k \preceq k \nabla h$.

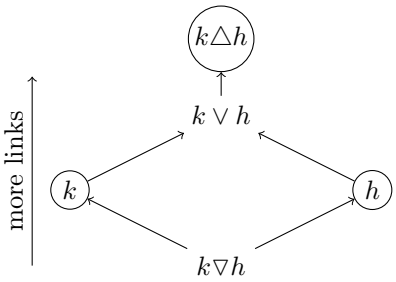

Figure 2: Relations between the disjunction, meet and join of two link key candidates k and h with respect to extended subsumption ($\preceq$). Circled nodes are link key candidates, edges represent subsumption, and outbound nodes are subsumees.

A disjunction of link key expressions only made up of link key candidates will be called disjunction of link key candidates. The set of disjunctions from a set of link key candidates K will be denoted by $K^\vee$. It is easy to prove that $\simeq$ is an equivalence relation over $K^\vee$. The quotient of K by $\simeq$ will be denoted by $K^\vee_\simeq$.

4.2 Quality measures

Existing quality measures for single link key candidates are defined on their generated links, so they can be straightforwardly extended to the case of disjunctions of link key candidates. Indeed, $\delta(k)$ and $\gamma(k)$ use L_k only, which has already been defined for disjunctions of link key expressions in Definition 4. Table 2 shows the values of δ, γ and κ for some of the link key candidates of Figure 1.

$k \vee h$	$\kappa(k)$	$\delta(k \vee h)$	$\kappa(k \vee h)$	$\gamma(k \vee h)$
$k_2 \vee k_6$	.77	.75	.86	1.0
$k_2 \vee k_8$	.77	.71	.77	.83
$k_8 \vee k_4$	.50	1.0	.91	.83
$k_4 \vee k_6$	.67	1.0	1.0	1.0
$k_8 \vee k_5$	.50	.54	.68	.92

Table 2: Values of the quality measures for the disjunctions of some of the link key candidates of Figure 1. They all improve coverage (γ); all, but the last one, have higher or equal harmonic mean (κ) than the best link key candidate of the lattice ($\kappa(k_2) = .77$); and $k_2 \vee k_8$ does not improve on k_2 as k_2 subsumes k_8.

Property 2 still holds for disjunctions of link key expressions and, in particular, the relations stated in Property 6.

Once the quality of a disjunction of link key candidates can be measured, the problem is to obtain the best disjunction(s).

5 Disjunction extraction

From the set K of all link key candidates returned by a link key candidate extraction algorithm, we address the extraction of the best disjunction(s) of link key candidates with respect to the harmonic mean of discriminability and coverage κ. If $|K| = n$, however, there are potentially 2^n disjunctions of link key candidates. In what follows, we propose two different strategies to search disjunctions of link key candidates efficiently. They both exploit antichains.

5.1 Exploiting antichains

For a given pair of data sets, the set of link key candidates with $\preceq$ (extended subsumption) form a lattice. The search of disjunctions of link key candidates can be restricted to the search of antichains [Garg 2015] of elements in this lattice (Property 7). Indeed, in our setting, antichains represent non redundant disjunctions of link key candidates. More formally, an antichain is a set of link key candidates $\{k_1, \ldots, k_m\}$ such that for every $i, j = 1, \ldots, m$ with $i \neq j$, neither $k_i \preceq k_j$ nor $k_j \preceq k_i$.[2] A disjunction of link key candidates can be

[2] An *antichain* of a partially ordered set is a subset of pairwise non comparable elements.

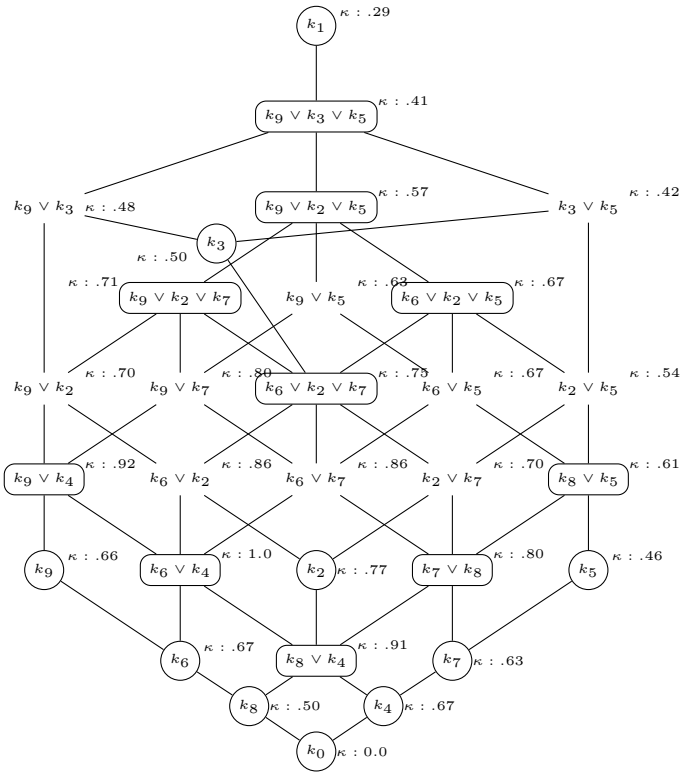

Figure 3: Full antichain lattice $\langle K^{\vee}_{\simeq}, \preceq \rangle$ from Example 3 including link key candidates (in circles), maximal antichains (in rounded boxes) and antichains (κ=harmonic mean between discriminability and coverage).

straightforwardly built from an antichain of link key candidates (remember that $\vee$ is associative and commutative).

Property 7 *Any disjunction of link key candidates is equivalent to one built from an antichain of the link key candidate lattice.*

If two link key candidates k and h are such that $k \preceq h$ then, by Property 4, $k \vee h \simeq k$. So any disjunction of link key candidates is equivalent to the disjunction of its non comparable link key candidates.

The number of antichains of a lattice is difficult to establish a priori [Garg 2015], the worst case being 2^n. In the case of the lattice of link key candidates of Figure 1, there are 10 candidates, thus 1024 possible disjunctions. However, there are only 12 maximal antichains covering 30 antichains. All these disjunctions of link keys can be organised in a lattice $\langle K^{\vee}_{\simeq}, \preceq \rangle$. The one corresponding to Example 3 is displayed in Figure 3.

Exploring the antichain lattice may be achieved in various ways but it reaches its limits fast if there are many link key candidates. Indeed, (1) the number of non maximal antichains may be daunting and (2) the best candidate in terms of κ may be anywhere in the lattice. Hence, we consider next how to deal with such problems in large data sets.

5.2 In search of the best antichain

We propose two strategies for searching the best antichains. These approaches are not exhaustive but use heuristics that may help to find good antichains.

The *top-k strategy* selects the top-k candidates according to some evaluation measure and then performs an exhaustive enumeration of antichains on this selection. This assumes that the best antichains are those which only contain the best link key candidates.

The *expand-best strategy* performs a best-first search. It explores the antichains from the best individual link key candidates and by iteratively replacing the best antichain by its expansion (i.e. the set of antichains obtained by adding another individual link key candidate). At each step, an antichain is selected only if it is better than those explored thus far. The process stops after x iterations without any improvement. It assumes that the better an antichain is, the more chances that it can produce better antichains.

Both approaches, with the data sets of Example 3, return quickly the best disjunction of link keys: $k_6 \vee k_4$. We compare these approaches experimentally in Section 6.

6 Experimental evaluation

In this section, we report on experiments with the two strategies for extracting disjunctions of link key candidates proposed in Section 5. Our aim was to test the following hypothesis: *Disjunctions of link key candidates generate better link sets, in terms of F-measure, than single link key candidates.*

To test the above hypothesis, we used OAEI data sets and two data sets from the library domain. The data sets of each OAEI task share the same ontology, but this is not the case of the two data sets of the library domain. We do not compare our results with the results of other data interlinking approaches, as it is not the goal of this evaluation. Besides, the results of the different OAEI campaigns are available online.[3]

For each OAEI task, we first generated single link key candidates with *Linkex*, our link key candidate extraction tool.[4] We set up Linkex to perform a basic normalisation of data values, and to deal with inverse and 2-length composition of properties. The data normalisation consisted in removing diacritics, tokenising strings and sorting the resulting bag of tokens.

Second, we applied the top-k and expand-best strategies for extracting the best disjunctions of link key candidates according to κ. For the top-k strategy, we chose $k = 10, 15, 20, 25, 30$. Since all these different values of k produced very similar results, we only present here the results of $k = 10$. For expand-best, we stopped the search after $x = 100$ iterations without finding a better antichain.

Finally, the comparison between disjunctions and single link key candidates was done by computing precision, recall and F-measure against the reference link sets.

Table 3 shows statistics of the used data sets and the extracted link key candidates. In particular, it shows the number of properties (named properties, inverse or composition of properties) that appear in at least one extracted candidate. Also, for each task, it shows the number of extracted link key candidates, and the precision, recall and F-measure of the candidate with the highest κ value.

Table 4 shows the results of the antichain extraction. For each task, it shows the precision, recall and F-measure of the antichains with the highest κ value obtained by the two strategies. For the top-10 strategy, it also shows the number of antichains extracted and the number of maximal antichains among them (e.g. 43 and 12 for Restaurants, respectively). For the expand-best strategy, it shows the number of generated antichains — *tested* column (e.g. 337 for Restaurants) — and also the position of the best antichain in the sequence of generated antichains — *best* column (e.g. 34 for Restaurants).

In the remainder of the section, we analyse the results and discuss the general lessons learnt from the experiments.

6.1 Simple data sets (OAEI 2010)

These tasks were performed using the OAEI 2010 data sets.[5] For all of them, the two strategies allowed to find an antichain with better F-measure than the best individual link key. The top-10 strategy performed better than the expand-best strategy on Restaurant and Person2. The expand-best strategy generated more antichains than the top-10 strategy. This was specially true for Person2 on which

[3] http://oaei.ontologymatching.org

[4] https://gitlab.inria.fr/moex/linkex.

[5] http://oaei.ontologymatching.org/2010/: Person1, Person2, Restaurants.

Data set	#inst.	#prop.	#triples	#cand	Prec.	F-meas.	Rec.
Restaurants1	113	4	1 130	20	0.477	0.58	0.741
Restaurants2	752	4	7 520				
Person11	500	9	9 000	613	1	0.974	0.95
Person12	500	10	7 000				
Person21	600	9	10 800	521	0.206	0.27	0.39
Person22	400	10	5 600				
PP-1	32	9	2 530	27	0.833	0.714	0.625
BnF-1	32	9	2 189				
PP-2	201	13	12 757	101	0.833	0.712	0.622
BnF-2	201	15	10 622				
PP-3	41	11	2 970	38	0.622	0.571	0.683
BnF-3	41	12	2 610				
Abox1	349	38	10 001	2 277	0.816	0.794	0.773
Abox2	284	58	10 022				
Abes	15 421	6	66 610	933	0.656	0.614	0.578
BnF	8 162	9	106 224				

Table 3: Data sets and extracted link key candidate statistics.

the expand-best strategy generated a large number of antichains. Moreover, the F-measure was particularly low. This is due to the fact that the δ measure is not well-adapted to this specific task since the first data set of Person2 contains a lot of redundancy.

6.2 Doremus (OAEI 2016)

The Doremus data sets of OAEI 2016 are small data sets — PP-n and BnF-n ($n = 1, 2, 3$) in Table 3 — from cultural institutions with different kinds of heterogeneity.[6] In this case, our hypothesis was clearly confirmed as both strategies allowed to gain at least 4 points of F-measure with respect to the best single link key candidate. Unlike the previous tasks (Section 6.1), expand-best outperformed top-10 on the three tasks. But, as for the previous tasks, expand-best generated more antichains than top-10.

6.3 SPIMBench (OAEI 2018)

We also applied our strategies on SPIMBench Sandbox from OAEI 2018.[7] These data sets (Abox1, Abox2) include around 380 instances and 10 000 triples. The goal is to find links between the instances of the classes NewsItem, BlogPost and Programme.

[6] http://islab.di.unimi.it/content/im_oaei/2016/

[7] http://oaei.ontologymatching.org/2018/spimbench.html

Task	Strategy	Prec.	F-meas.	Rec.	time	#a.c /tested	#max a.c /best
Restaurants	top-10	0.483	0.596	0.777	<1"	43	12
	expand-best	0.481	0.594	0.777	<1"	337	34
Person1	top-10	1	1	1	<1"	223	5
	expand-best	1	1	1	<3"	1041	901
Person2	top-10	0.348	0.425	0.545	<1"	311	8
	expand-best	0.265	0.369	0.608	<3"	30 110	18 523
Doremus 1	top-10	0.793	0.754	0.719	<1"	72	9
	expand-best	0.806	0.794	0.781	<1"	326	54
Doremus 2	top-10	0.829	0.799	0.771	<1"	219	9
	expand-best	0.830	0.802	0.776	<1"	2187	420
Doremus 3	top-10	0.569	0.667	0.805	<1"	140	9
	expand-best	0.596	0.694	0.829	<1"	416	82
SPIMBench	top-10	0.816	0.794	0.773	4"	47	12
	expand-best	0.805	0.788	0.773	1'20"	26 557	3 318
Libraries	top-10	0.563	0.616	0.679	<1"	134	16
	expand-best	0.363	0.474	0.681	42"	65 112	35 193

Table 4: Results.

In this case, the number of extracted link key candidates was quite large. This is due to the high number of properties of the data sets. Thus, an exhaustive search of all antichains was not feasible.

The best link key candidate is already of a high quality (≈ 0.8) and neither of the two strategies was able to find a better antichain. Due to the heuristic nature of the proposed strategies, it is unclear whether this result is the consequence of the non existence of a better link key or the incompleteness of the procedure.

6.4 Libraries

For this last task, we used sample data sets provided by two French libraries: the "Bibliothèque Nationale de France" (BnF),[8] and the "Agence Bibliographique de l'Enseignement Supérieur" (Abes).[9] The sampling consisted in extracting within each data set the authors that have one of the top-1000 most common homonym names (name and first name). The books written by each author are available. The classes to link are the ones representing authors. Unlike the OAEI data sets, these data sets use different ontologies. Only a partial reference link set was available.

The top-10 strategy marginally improved the F-measure of the best link key candidate, while expand-best had a lower score. Both strategies improved recall but precision was negatively impacted. This may be due to the fact that the results were evaluated against a partial reference. The best disjunction

[8] https://data.bnf.fr/

[9] https://www.idref.fr/

generated by top-10 contains 2 link keys while the one generated by expand-best contains 62. This last disjunction contains many link keys with very low coverage: 10 only generate one link and 36 generate less than ten links. This goes against the idea that link keys are general linking conditions and that they always cover most of the instances to link.

6.5 General remarks

Overall, our hypothesis was confirmed: disjunctions of link keys bring an improvement to data interlinking with respect to single link keys.

The experimental results show that the top-10 strategy always allows to find a disjunction better than the best single link key candidate. In addition, the expand-best strategy always generates longer disjunctions than the top-10 strategy. Indeed, whereas the top-10 strategy generated disjunctions of only two or three link key candidates in all cases, the expand-best strategy generated very long disjunctions in some cases: 22 for Person2 and 10 for the Doremus2. Consequently, the expand-best strategy favours recall over precision. Furthermore, top-10 scales better than expand-best.

The proposed κ measure is not optimal in the sense that some generated link key candidates are better in terms of F-measure than those selected by κ. This is especially true if the data sets are very different in size (number of instances) and when the target link set is far from a one-to-one mapping. Further work will be needed for identifying more suitable measures.

Concerning the OAEI tasks, we must admit that link keys (single or disjunction candidates) do not provide the best results of the campaigns on the first three tasks. However, these data sets should be considered as easier to be worked with by the other existing data interlinking approaches, as they use the same ontologies, which is not a requirement for link keys. Indeed, Linkex always considers as different the properties of two different datasets. Only the pair of classes whose instances are to be linked is given by input.

7 Conclusion

We have defined disjunction of link keys, and extended link key candidate extraction to deal with disjunctions of link key candidates.

The added value of the provided extraction approach, compared to other existing approaches, is that it does not rely on supervised machine learning, and, hence, does not need training links, nor it requires an input alignment between the ontologies of the data sets to interlink. Given the size of the search space, we introduced heuristics to extract disjunctions of link key candidates.

We have shown through an experimental evaluation that the extracted disjunctions improve the F-measure of individual link key candidates. However, the experiments demonstrate that further improvements are needed, either by combining both heuristics, or by developing more suitable unsupervised measures.

Acknowledgements

This work has been partially supported by the ANR project Elker (ANR-17-CE23-0007-01).

References

[Achichi et al. 2016] Manel Achichi, Mohamed Ben Ellefi, Danai Symeonidou, and Konstantin Todorov. 2016. Automatic key selection for data linking. In *Knowledge Engineering and Knowledge Management - 20th International Conference, EKAW 2016, Bologna, Italy, November 19-23, 2016, Proceedings (LNCS)*, Vol. 10024. 3–18.

[Al-Bakri et al. 2016] Mustafa Al-Bakri, Manuel Atencia, Jérôme David, Steffen Lalande, and Marie-Christine Rousset. 2016. Uncertainty-sensitive reasoning for inferring sameAs facts in linked data. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016) (Frontiers in Artificial Intelligence and Applications)*, Vol. 285. IOS Press, 698–706.

[Al-Bakri et al. 2015] Mustafa Al-Bakri, Manuel Atencia, Steffen Lalande, and Marie-Christine Rousset. 2015. Inferring same-as facts from Linked Data: an iterative import-by-query approach. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*. AAAI Press, 9–15.

[Atencia et al. 2014] Manuel Atencia, Jérôme David, and Jérôme Euzenat. 2014. Data interlinking through robust linkkey extraction. In *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014) (Frontiers in Artificial Intelligence and Applications)*, Vol. 263. IOS Press, 15–20.

[Atencia et al. 2019] Manuel Atencia, Jérôme David, Jérôme Euzenat, Amedeo Napoli, and Jérémy Vizzini. 2019. Link key candidate extraction with relational concept analysis. *Discrete applied mathematics* (2019). https://dx.doi.org/10.1016/j.dam.2019.02.012

[Atencia et al. 2012] Manuel Atencia, Jérôme David, and François Scharffe. 2012. Keys and pseudo-keys detection for web datasets cleansing and interlinking. In *Knowledge Engineering and Knowledge Management - 18th International Conference, EKAW 2012, Galway City, Ireland, October 8-12, 2012. Proceedings (LNCS)*, Vol. 7603. Springer, 144–153.

[Euzenat and Shvaiko 2013] Jérôme Euzenat and Pavel Shvaiko. 2013. *Ontology matching* (2nd ed.). Springer, Heidelberg (DE).

[Farah et al. 2017] Houssameddine Farah, Danai Symeonidou, and Konstantin Todorov. 2017. KeyRanker: Automatic RDF key ranking for data linking. In *Proceedings of the Knowledge Capture Conference, K-CAP 2017, Austin, TX, USA, December 4-6, 2017*. ACM, 7:1–7:8.

[Ferrara et al. 2011] Alfio Ferrara, Andriy Nikolov, and François Scharffe. 2011. Data Linking for the Semantic Web. *International Journal of Semantic Web and Information Systems* 7, 3 (2011), 46–76.

[Ganter and Wille 1999] Bernhard Ganter and Rudolf Wille. 1999. *Formal Concept Analysis*. Springer, Berlin, DE.

[Garg 2015] Vijay Garg. 2015. *Introduction to lattice theory with computer science applications*. John Wiley and sons.

[Hogan et al. 2012] Aidan Hogan, Antoine Zimmermann, Jürgen Umbrich, Axel Polleres, and Stefan Decker. 2012. Scalable and distributed methods for entity matching, consolidation and disambiguation over linked data corpora. *Journal of Web Semantics* 10 (2012), 76–110.

[Isele and Bizer 2013] Robert Isele and Christian Bizer. 2013. Active learning of expressive linkage rules using genetic programming. *Journal of Web Semantics* 23 (2013), 2–15.

[Nentwig et al. 2017] Markus Nentwig, Michael Hartung, Axel-Cyrille Ngonga Ngomo, and Erhard Rahm. 2017. A survey of current link discovery frameworks. *Semantic Web* 8, 3 (2017), 419–436.

[Ngonga Ngomo and Auer 2011] Axel-Cyrille Ngonga Ngomo and Sören Auer. 2011. LIMES: A time-efficient approach for large-scale link discovery on the Web of Data. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*. IJCAI/AAAI, 2312–2317.

[Saïs et al. 2007] Fatiha Saïs, Nathalie Pernelle, and Marie-Christine Rousset. 2007. L2R: A Logical Method for Reference Reconciliation. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*. AAAI Press, 329–334.

[Sherif et al. 2017] Mohamed Ahmed Sherif, Axel-Cyrille Ngonga Ngomo, and Jens Lehmann. 2017. Wombat - A generalization approach for automatic link discovery. In *The Semantic Web - 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 - June 1, 2017, Proceedings, Part I (LNCS)*, Vol. 10249. Springer, 103–119.

[Symeonidou et al. 2014] Danai Symeonidou, Vincent Armant, Nathalie Pernelle, and Fatiha Saïs. 2014. SAKey: Scalable Almost Key Discovery in RDF Data. In *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, (LNCS)*, Vol. 8796. Springer, 33–49.

[Volz et al. 2009] Julius Volz, Christian Bizer, Martin Gaedke, and Georgi Kobilarov. 2009. Silk – A link discovery framework for the Web of Data. In *Proceedings of the WWW2009 Workshop on Linked Data on the Web, LDOW 2009, Madrid, Spain, April 20, 2009. (CEUR Workshop Proceedings)*, Vol. 538. CEUR-WS.org.